Advance Learning Reductions Day

John Langford (With help from many others)

Yahoo Research

Helsinki, Finland, Oct. 13

Idea for the day: cover several advanced learning reductions

- Advanced applications are where reductions are most useful, empirically.

Much of this is new, nearly new, or not yet done research.

Outline:

1. Reinforcement Learning via Reduction
   - With application to Structured Prediction
2. Ranking
3. Open Problems/Current Work

How do we think about a general RL problem?

$D(o', r|(o,a,r)^*, o, a)$ = conditional probability table

1. $o', o \in O$ = observations

2. $r \in [0, \infty)$ = reward

3. $a \in A$ = action:

Find $\pi((o,a,r)^*, o) \rightarrow a$ maximizing:

$$\eta(\pi) = E_{(o,a,r)^T \sim \pi, D}\left[\sum_{t=1}^{T} r_t\right]$$

An "Algorithm"

Let π^*= optimal policy

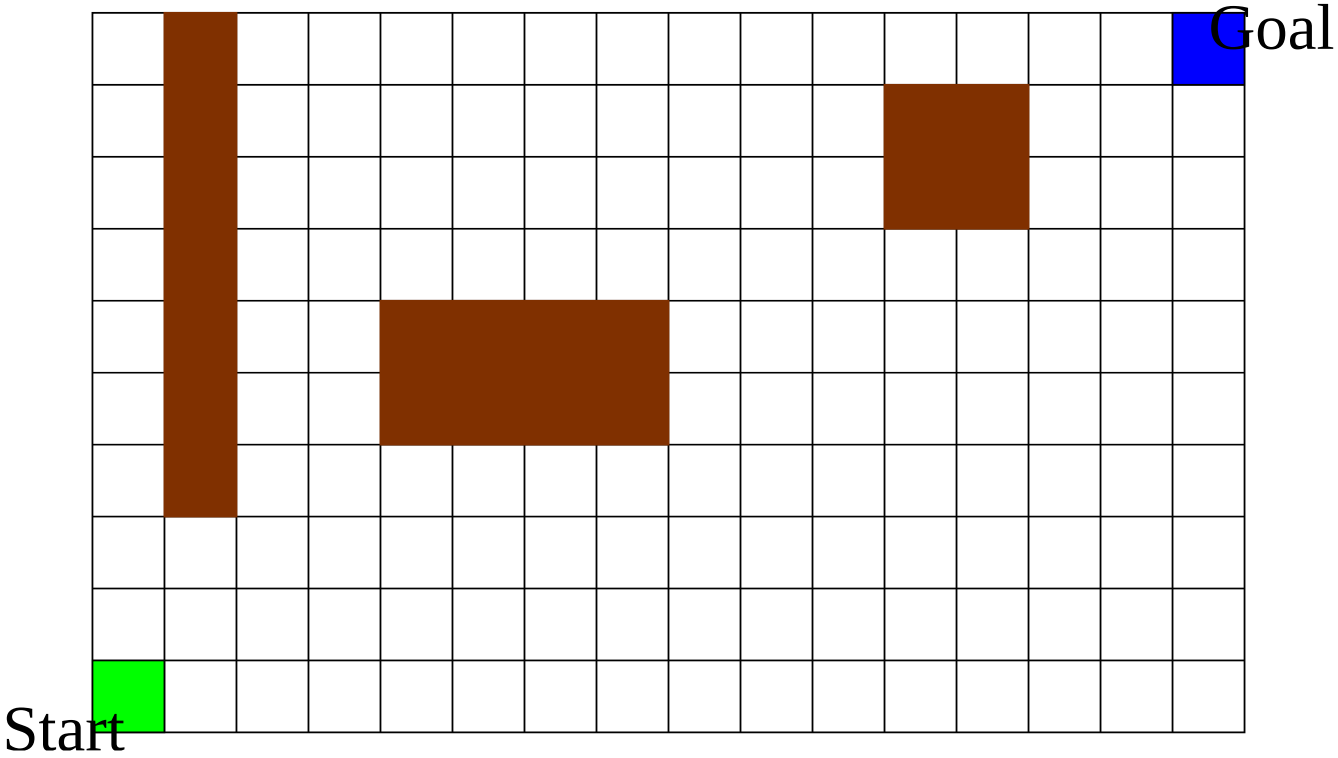
1. Given empty history, predict optimal a assuming π^* followed for $T-1$ timesteps afterwards.

2. Given first prediction, 1 step history, predict optimal a assuming π^* followed for $T-2$ timesteps afterwards.

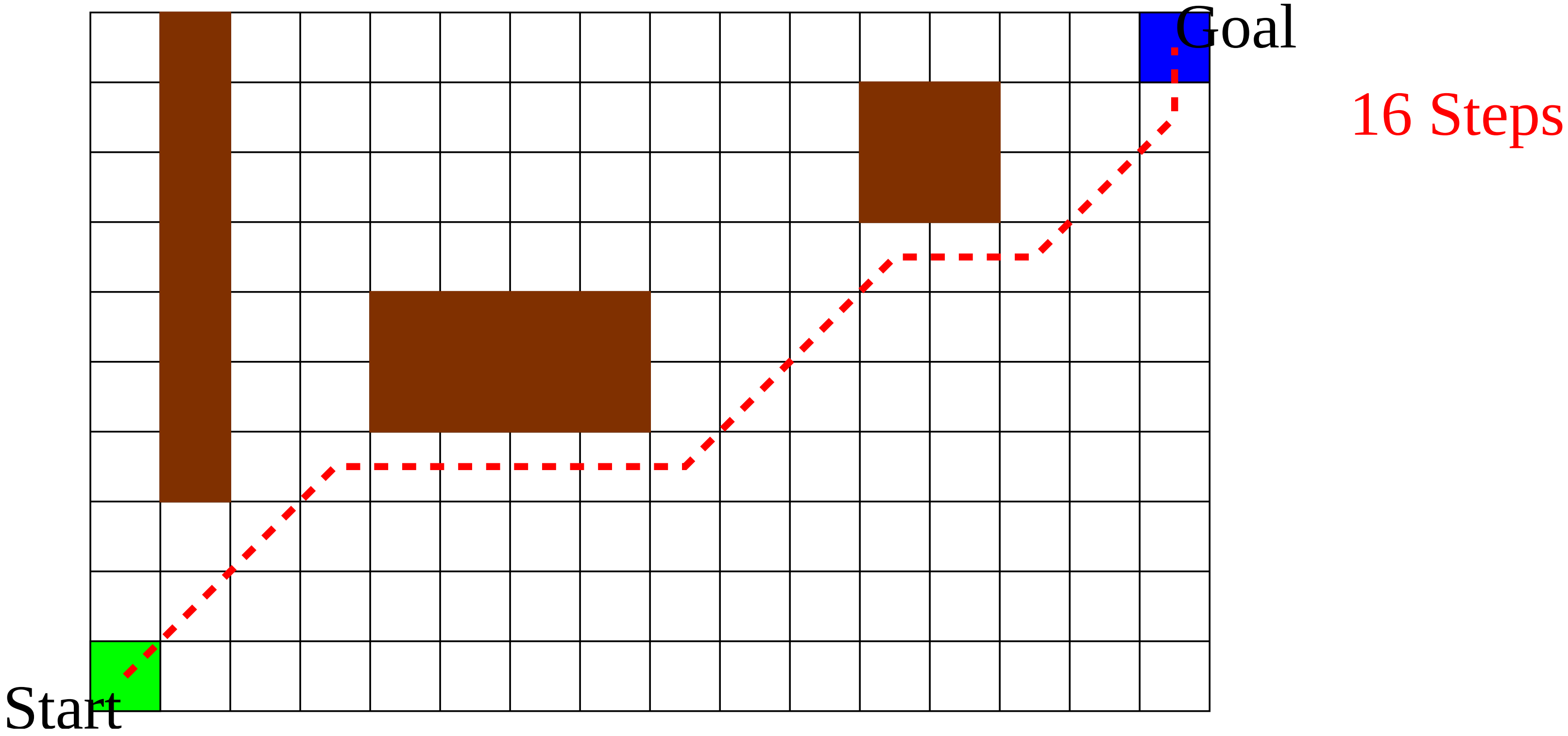
3. Given 2 step history, predict optimal a assuming π^* followed for $T-3$ timesteps afterwards.

4. ...

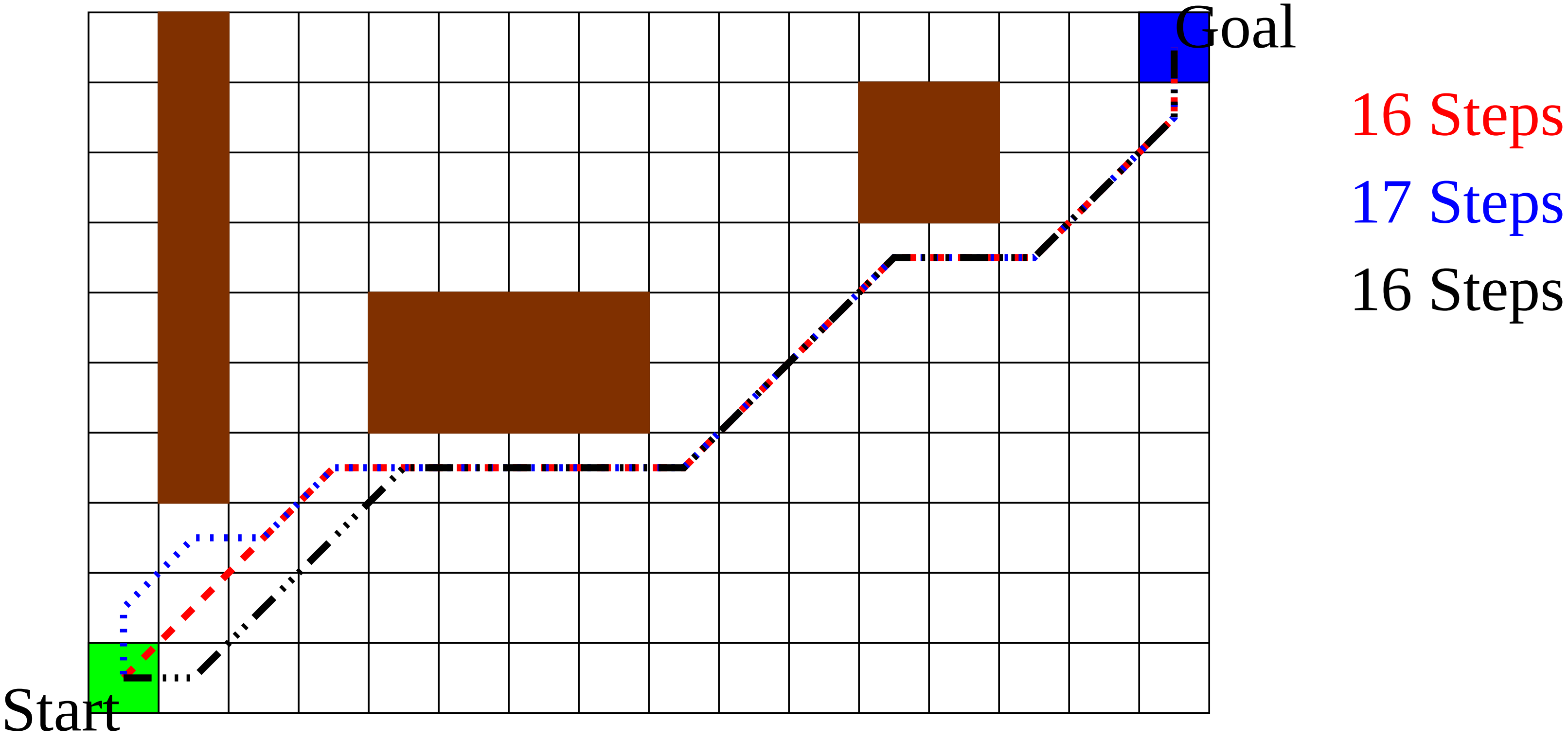
The Algorithm: Pretty Pictures



The Algorithm: Pretty Pictures

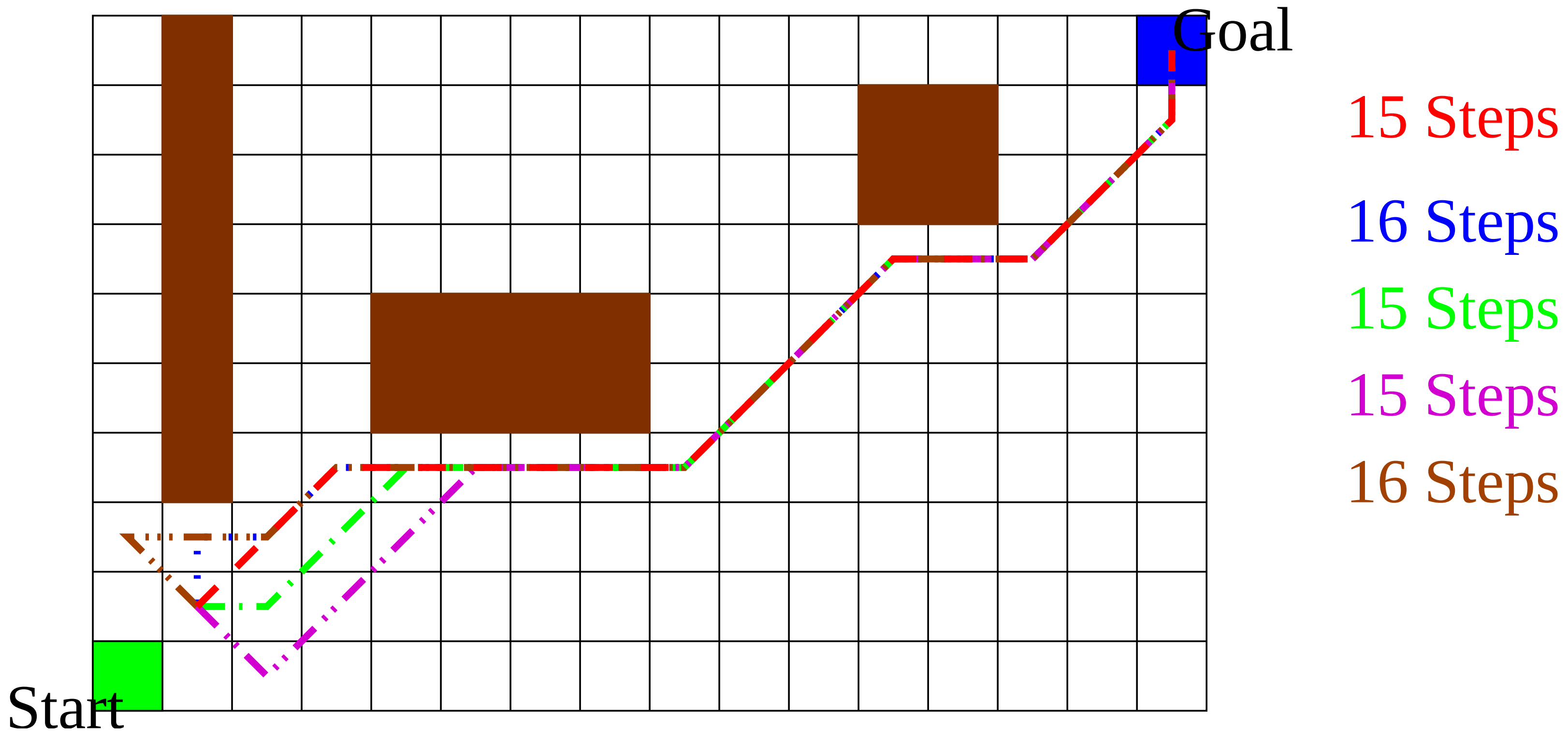


The Algorithm: Pretty Pictures



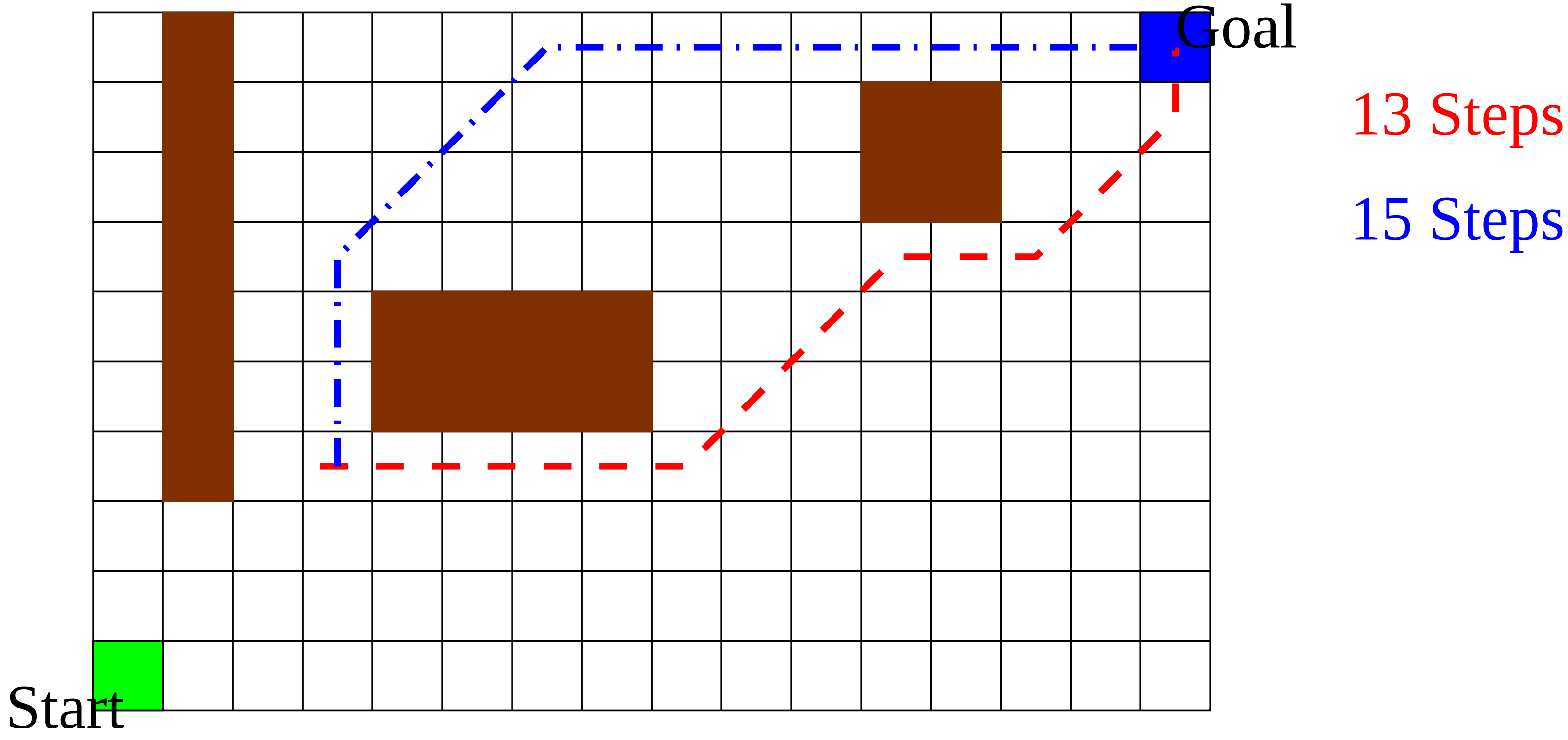
Cost for action $a = \text{steps}(a) - \min_{a'} \text{steps}(a') =$ 0, 1, 0

The Algorithm: Pretty Pictures



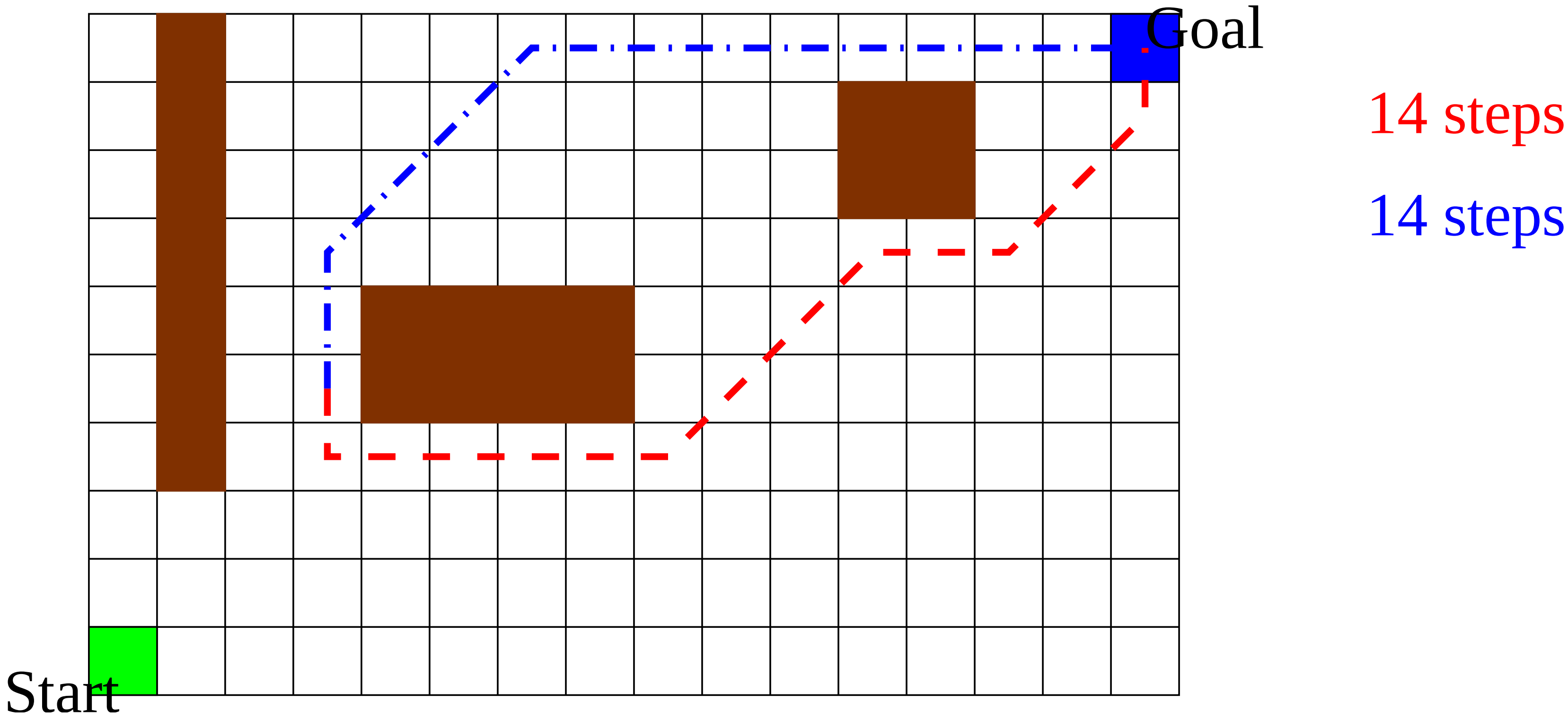
cost = 0, 1, 0, 0, 1

The Algorithm: Pretty Pictures



Cost = 0, 2

The Algorithm: Pretty Pictures



Cost = 0, 0 Note: condition on suboptimal outcome at previous step.

The Critical Lemma

$A_t^*(D,\pi,h_t) = E_{(a,o,r)^T \;\; D,(\pi,h_t,\pi^*)}\left[\sum_{t'=t}^{T} r_{t'}\right]$ = Expected reward sum when starting with π, acting according to h_t and completing with the optimal policy. (A for "Advantage".)

Lemma: For all D, $\pi = (h_1, ..., h_T)$,

$$\eta(\pi^*) - \eta(\pi) = \sum_{t=1}^{T} A_t^*(D,\pi,\pi_t^*) - A_t^*(D,\pi,h_t)$$

= sum of cost sensitive losses.

Proof

$$\eta(\pi^*) - \eta(\pi) = E_{(o,a,r)^T \sim D,\pi^*}\left[\sum_{t=1}^{T} r_t\right] - E_{(o,a,r)^T \sim D,\pi}\left[\sum_{t=1}^{T} r_t\right]$$

$$= \sum_{t'=1}^{T} E_{(o,a,r)^T \sim D,(\pi,\pi_{t'}^*,\pi^*)}\left[\sum_{t=1}^{T} r_t\right] - E_{(o,a,r)^T \sim D,(\pi,\pi_{t'},\pi^*)}\left[\sum_{t=1}^{T} r_t\right]$$

$$= \sum_{t'=1}^{T} E_{(o,a,r)^T \sim D,(\pi,\pi_{t'}^*,\pi^*)}\left[\sum_{t=t'}^{T} r_t\right] - E_{(o,a,r)^T \sim D,(\pi,h_{t'},\pi^*)}\left[\sum_{t=t'}^{T} r_t\right]$$

$$= \sum_{t=1}^{T} A_t^*(D,\pi,\pi_t^*) - A_t^*(D,\pi,h_t)$$

Ways to think about the result

1. Reduction of RL to classification given oracle access to optimal policy. (A form of "apprenticeship learning".) [Surprisingly useful.]

2. A *nonconstructive* reduction of RL to classification. There exists a set of cost sensitive classification problems such that if they are solved well, we solve RL. (But which problem is unclear.)

What can we do constructively?

Given side information, we can subvert the exploration problem.

1. Generative model (GM) (high exponential)

2. Training time Acces to Optimal Policy + GM

3. State Distribution of (near) optimal policy + GM

Sparse Sampling

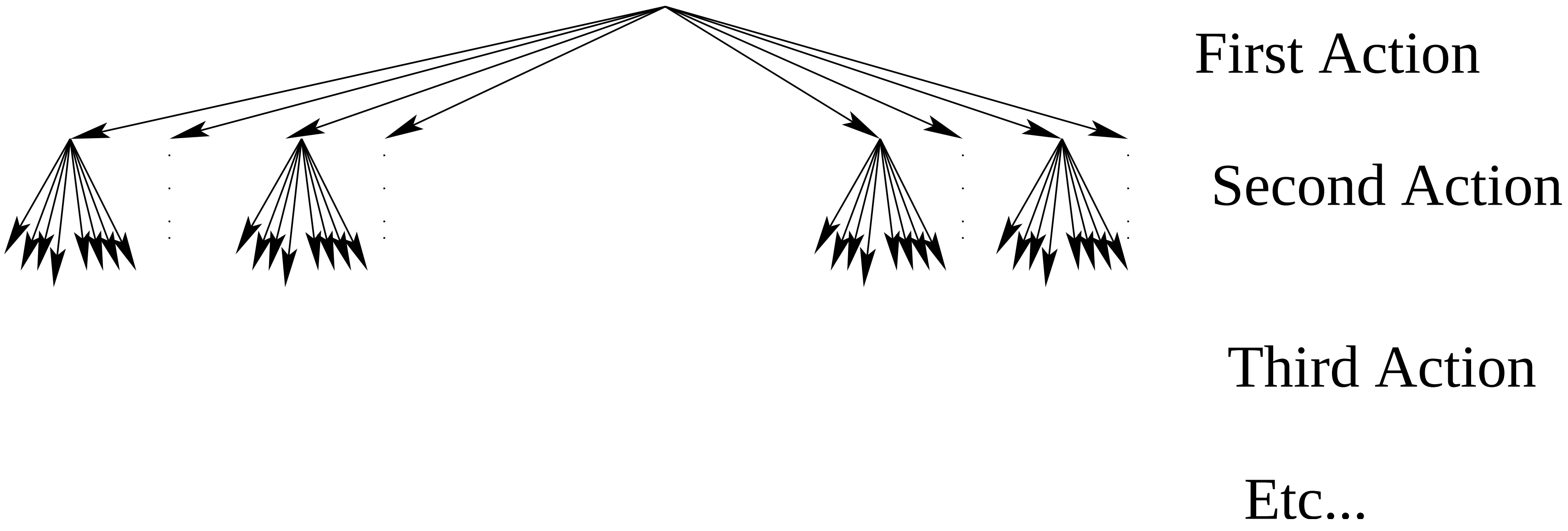
1. For every first action, sample k times

2. For every result, for every second action, sample k times.

3. For every result, for every third action, sample k times.

4. etc..

Sparse Sampling, the picture



Picture is difficult, very high sample complexity.

Sparse Sampling Decisions

1. For each action, expectation over samples at step T

2. Max over actions.

3. For each action, expectation over T maxes at step $T-1$.

4. Etc...

$\Rightarrow$ Estimate of $A_t(D, \pi, h_t)$ accurate to $(T-t)\sqrt{\frac{\log A + \log \frac{1}{\delta}}{k}}$.

$\Rightarrow$ can apply reduction

What can we do constructively?

Given side information, we can subvert the exploration problem.

1. Generative model (GM) (high exponential)

2. Training time Acces to Optimal Policy + GM

3. State Distribution of (near) optimal policy + GM

Training time Acces to Optimal Policy + GM

Algorithm from before applies.

But: Learning new classifier for every timestep is computationally difficult.

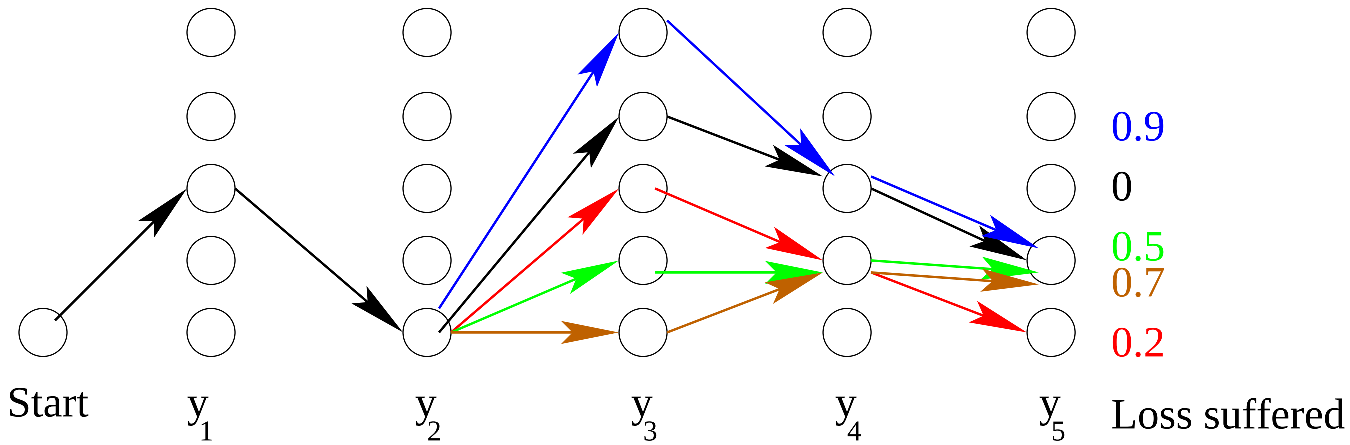
Solution: Learn one classifier for all timesteps simultaneously.

But: This breaks good reduction properties.

Solution: Use "Conservative Policy Iteration" trick.

The Conservative Policy Iteration trick

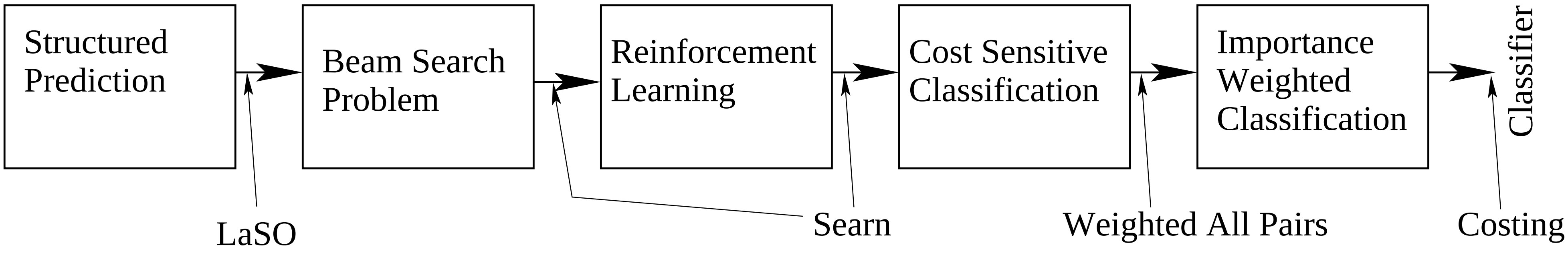
We have policy π_{i-1} which we use to define examples for h using GM:



Iteration = $\pi_i \leftarrow (1-\beta)\pi_{i-1} + \beta h$ (stochastic interpolation)

If you start with π^*, after $O\left(\frac{1}{\beta}\right)$ iterations, the policy is entirely learned.

Searn (= Search + Learn)



(The experiments were run with “weighted all pairs”, “costing”, and {Perceptron, SVM}.)

Cost-sensitive classification: Multiclass except each choice has a different loss. The goal is to minimize expected loss:

$$\arg\min_h E_{x,c^k \sim D}[c_{h(x)}]$$

The Searn Theorem

Theorem: For *all* RL problems D, for *all* sequences of learned classifiers $c_1, c_2, \ldots$, for small β, after $2T^3 \ln T$ iterations,

$$\text{Loss(learned policy)}$$

$$\leq \text{Loss(Initial Policy)} + \text{Average classifier loss} * 2T \ln T + (1 + \ln T)\frac{\text{max loss}}{T}$$

Note: This is actually *worse* than $O(T)$ steps. However, in practice, line search over β implies $< T$ classifiers learned.

Searn Experimental Results

Spanish Named Entity Recognition (1517 test)

	CRF(300)	SVM(300)	SVMISO(300)	Searn(300)	Searn(8324)
Acc.	94.83	94.94	94.9	95.01	97.67

(Multidigit) Handwriting Recognition (5500 or 600 test)

Training set size	Perceptron	Logistic	SVM	M^3N	Searn
600 train	65.56	68.65	82.63	87.5	88.2
5500 train	70.05	72.10	82.52	?	90.91

Joint sequence labeling (8936 train, 2012 test)

	Inc. P.	LASO	Factored CRF	CRF	Searn
Joint Accuracy	93.12	95.12	96.48	?	96.81
Chunk F-score	92.44	93.49	93.87	94.77	94.36
Parts of speech	98.05	98.90	98.92	?	99.07

Document Summarization (fraction of max "Rouge-1+2")

word count	DUC 05 winner	BD03	BD05	Searn	Searn (word)
100	?	0.803	0.900	0.971	1.129
250	0.749	0.696	0.883	0.906	1.170

Viewpoints on Searn

- A structured prediction algorithm which decomposes global loss into local loss.

- A system for general purpose integration of search and learning.

- A reinforcement learning algorithm which operates given access to a (near) optimal policy + simulator.

What can we do constructively?

Given side information, we can subvert the exploration problem.

1. Generative model (GM) (high exponential)

2. Training time Acces to Optimal Policy + GM

3. State Distribution of (near) optimal policy + GM

State Distribution of (near) optimal policy + GM

What if (near) optimal policy is missing?

Sometimes we only know where an optimal policy visits. Is this enough?

The PSDP ("Policy Search by Dynamic Programming") algorithm

Let D_1, ..., D_T = distribution over states at timestep t induced by π^*.

1. Repeatedly draw s from D_T and use GM to estimate rewards. Learn a Tth step policy π_T via cost sensitive classification.

2. Repeatedly draw s from D_{T-1} and use GM + π_T to estimate rewards. Learn $(T-1)$th step policy π_{T-1} via cost sensitive classification.

3. etc...

The PSDP Theorem

$A_t^{\pi}(D_t, h_t) = E_{(a,o,r)^T \ \ D_t,(h_t,\pi)} \left[\sum_{t'=t}^{T} r_{t'} \right]$

(Like A_t^* except we *start* by assuming optimal policy and *end* with learned policy.)

Theorem: For all π^* generating $D_1, ..., D_T$, and all $\pi = (h_1, ..., h_T)$,

$$\eta(\pi^*) - \eta(\pi) = \sum_{t=1}^{T} A_t(D_t, \pi_t^*) - A_t(D_t, h_t)$$

$=$ sum of cost sensitive losses/rewards.

- The same as given $\pi^* + \text{GM}$ but Searn-style variant unknown.

Outline:

1. Reinforcement Learning via Reduction
    - With application to Structured Prediction
2. Ranking
3. Open Problems/Current Work

A Ranking problem: How do you produce a good ordering of elements?

Many different success measures.

- AUC = Area Under the receiver operating characteristic Curve = normalized bubble sort distance to correct order. (bubble sort distance = pairwise adjacent swap count)

- DCG = Discounted Cumulative Gain = $\sum g_i v_i$ where g_i = value at position i and v_i value of ith ranked element.

- etc...

A common approach and its problems

1. Use squared error regression to predict value $f(x)$.

2. Order elements according to $f(x)$.

Problem: Bad regret transform, $r \rightarrow nr$ where $n =$ number of elements.

Alternative Approach: the DEGREE algorithm

Train classifier c to predict “Is element x_1 larger than element x_2?” using all mixed pairs.

Test by ordering according to $|\{x_i : c(x, x_i) = 1\}|$ = number of wins

A theorem for the DEGREE algorithm

Theorem: For all ranking problems D, for all learned pairwise classifiers

$$r_{\text{AUC}}(\text{DEGREE}(\cdot, c), D) \leq 2r(c, \text{DEGREE}(D))$$

Note: no assumption of independent problems. D is over distributions producing a full training set.

Cool strange insight: This is the same guarantee that solving "minimum feedback arcset" (=NP hard problem) can give you.

Proof sketch

1. The AUC regret can be decomposed into a sum of *adjacent* element binary regrets

2. Let $r(i,j)$ be the regret of misordering elements i and j. For any i,j,k, $r(i,j)+r(j,k)=r(i,k)$

3. (2) implies binary regret can be decomposed into *adjacent* element binary regrets.

4. For each pair of adjacent elements, bound the ratio of (3) to (1). (highly technical)

Some thoughts

1. This doesn't create information from nothing: you may still have difficulty learning. The theorem *does* imply a large pairwise test error on mixed pairs exists. (Not so for regression.)

2. Importance weighted regression is not ruled out. Several recent papers have suggested this approach.

3. There are many other ranking losses which AUC should not be confused with.

4. The reduction also works with respect to a mixed pair source.

Outline:

1. Reinforcement Learning via Reduction
    - With application to Structured Prediction
2. Ranking
3. Open Problems/Current Work

Multiclass problem: The nasty $\sum$

Observation:

1. All cost sensitive classification reductions depend upon $\sum$ of costs.

2. Multiclass classification is independent of class count.

3. Multiclass classification as cost sensitive classification has cost sum $k - 1$.

Type: Technical, Finished

Solution (thanks to Pradeep Ravikumar)

1. Pick one of the classes with zero cost.

2. Create importance weighted multiclass problem: "Which class was picked?"

3. Reduce Importance weighted multiclass to multiclass, and multiclass to binary.

New dependence: replace $\sum$ with $\max_i c_i - \min_i c_i$.

Multiclass problem: The evil $\sqrt{\text{square root}}$

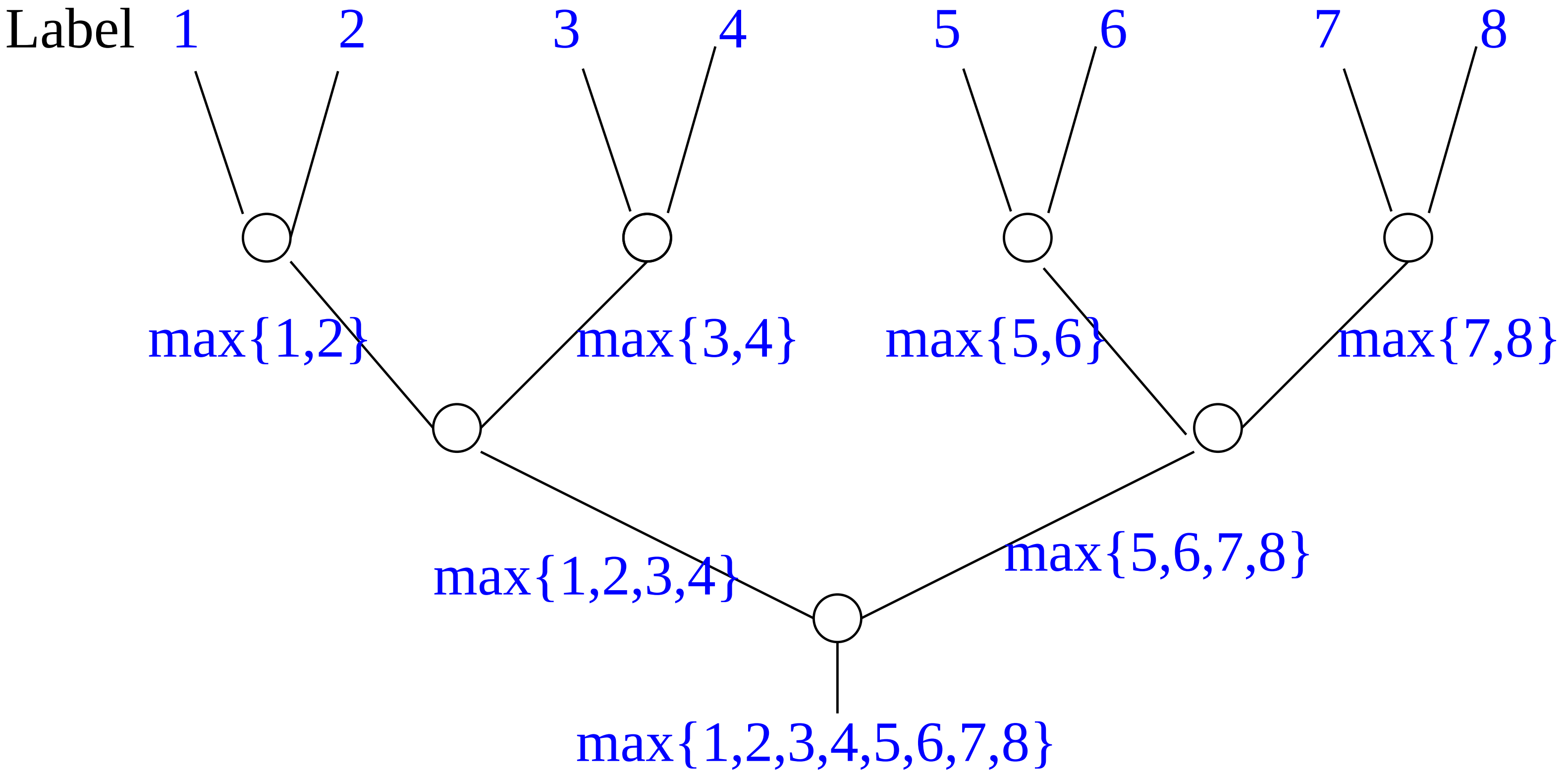
1. The reduction with the best dependence on k (constant) depends on $\sqrt{\text{binary regret}}$

2. The reduction with the best dependence (linear) on binary regret (the all pairs algorithm) depends on $k-1$.

Type: Technical/Algorithmic, in progress

A little bit of progress

A single elimination tournament satisfies $r_{\text{multi}} \leq r_{\text{binary}} \lceil \log k \rceil$.



Multiclass problem: Class count knowledge elimination

Some approaches for multiclass→binary depend upon knowledge and representatives of each class.

Some multiclass applications have a variable number of classes.

Can you achieve all the same guarantees with a variable number of classes that can be achieved with a fix number of classes?

Type: Technical or Algorithmic or Conceptual, Open

Lower Bounds

What is possible in a reductions framework?

1. How do you think about and prove lower bounds?

2. Are there unsolvable problems?

3. Are there limits on how tight the results can be? (The implications of boosting are interesting here.)

Matti is a good person to talk to here.

Type: Conceptual, Open

Reductions for Odd Data Sources

Often data sources are wrong.

For example:

1. You want to predict the best order to rank things in for a search engine.

2. But you mostly just know what people clicked on.

How can you systematically deal with these mismatches?

Type: Conceptual, Open

The Exploration Problem

No reductions deal with the exploration problem. Example:

1. There are many arms in front of you, and each has some feature values.

2. You pull one of them, and get a reward.

3. The feature values reset, and you go to (1).

Does a learning oracle help you find the right arm to pull?

Type: Conceptual, Open

New Reductions Frameworks

1. (Impagliazzo) A reduction R must satisfy: there exists R^{-1} such that $R(R^{-1})$ =identity (as applied to data or as applied to distributions over data).

2. (Osindero) A reduction R should commute with Bayes Law.

3. (Srebro) A reduction R should satisfy: for all binary predictors, there exists multiclass predictors such that binary predictor is a solution to multiclass predictor under reduction (and similar ideas).

Type: Conceptual, Open

Implementation

It is ridiculous that we don't yet have a library of reductions in a standard format.

1. Reductions are designed to be easy to code.

2. Reductions are designed to be put together in pieces.

Type: Research Engineering, in Progress. Talk to Bianca Zadrozny. Also see Hal's Searn code.

[Sparse Sampling] Michael Kearns, Yishay Mansour, and Andrew Ng, “A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes”, IJCAI 1999.

[CPI] Sham Kakade, John Langford “Approximately Optimal Approximate Reinforcement Learning”, ICML 2002.

[Many things] Sham Kakade, On the Sample Complexity of Reinforcement Learning Thesis Gatsby, UCL, 2003.

[PSDP] Drew Bagnell, Sham Kakade, Andrew Ng, & Jeff Schneider, “Policy Search by Dynamic Programming”, NIPS 2003.

[Given optimal Policy] John Langford and Bianca Zadrozny, “Relating Reinforcement Learning Performance to Classification Performance”, ICML 2005.

[Searn] Hal Daume III, John Langford, and Daniel Marcu, "Search-Based Structured Prediction" (unpublished).

[Ranking] N. Bansal, A. Beygelzimer, D. Coppersmith, J. Langford, G. Sorkin, "Robust Reductions of Ranking to Classification" (unpublished).

[Tournaments] A. Beygelzimer, J. Langford, and P. Ravikumar "Multiclass Classification with Inverted Trees" (unpublished)

Extra Discussion on many things, including RL, Reductions, etc...

http://hunch.net